



Методи чисельної оптимізації

Лекція 5: Градієнтний спуск та імпульс

Кочура Юрій Петрович
iuriy.kochura@gmail.com
[@y_kochura](https://twitter.com/y_kochura)

Сьогодні

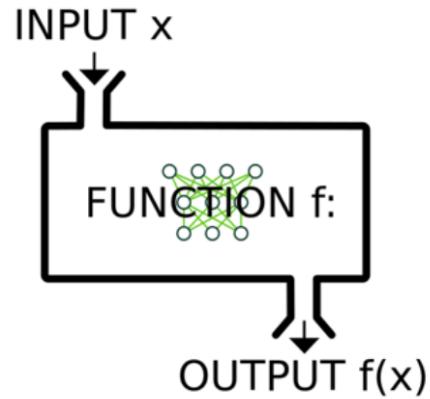
Як ефективно оптимізувати параметри?

- 🎙 Пакетний градієнтний спуск
- 🎙 Стохастичний градієнтний спуск
- 🎙 Міні-пакетний градієнтний спуск
- 🎙 Імпульс

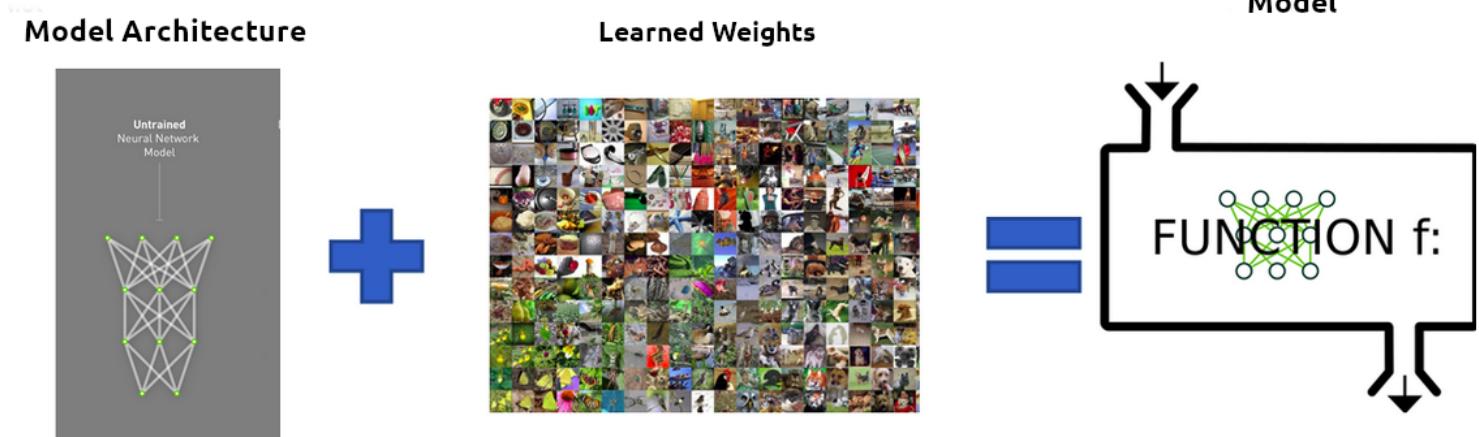
Оптимізація параметрів

Модель

Хоча те, що знаходиться всередині глибокої нейронної мережі, може бути складним, у своїй основі це просто функції. Вони приймають деякі вхідні дані та генерують деякі вихідні дані (прогнози).



Компоненти моделі



Загальний процес навчання для нейронних мереж

1. Визначте завдання + зберіть дані
2. Ініціалізуйте параметри
3. Оберіть алгоритм оптимізації
4. Повторіть ці кроки:
 - 4.1. Пряме поширення вхідних даних
 - 4.2. Обчислення цільової функцію витрат
 - 4.3. Зворотне поширення: обчисліть градієнти цільової функції втрат відносно параметрів
 - 4.4. Оновіть кожен параметр за допомогою градієнтів відповідно до алгоритму оптимізації

Optimization



Оптимізаційні алгоритми

Гіперпараметри & Параметри

- Результатом оптимізації є набір параметрів
- Гіперпараметри vs. Параметри

Гіперпараметри

- Коефіцієнт поділу датасету навчання/валідація/тестування
- Швидкість навчання в алгоритмах оптимізації
- Кількість прихованих шарів у НМ
- Кількість блоків активації в кожному шарі
- Розмір пакету

⋮

Параметри

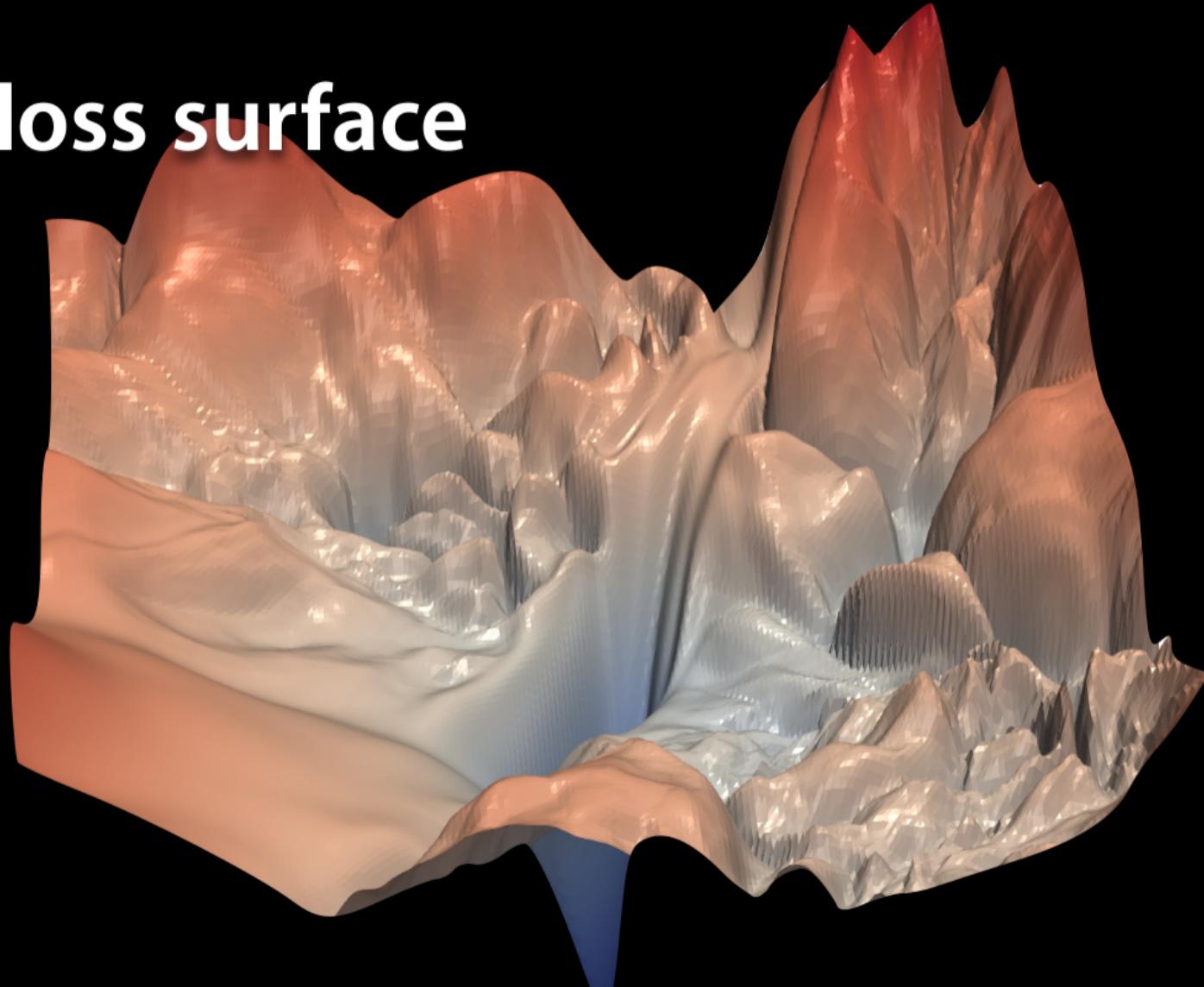
- Ваги та зсуви НМ
- Центроїди кластерів у кластеризації

Задача оптимізації

Мінімізація емпіричного ризику (втрат)

$$W_*^{\mathbf{d}} = \arg \min_W \mathcal{J}(W) = \arg \min_W \frac{1}{n} \sum_{i=1}^n \mathcal{L} \left(y^{(i)}, f(\mathbf{x}^{(i)}, W) \right)$$

loss surface



Практичні рекомендації

Навчання масивної глибокої нейронної мережі є тривалим та складним процесом.

Першим кроком до розуміння, налагодження та оптимізації нейронних мереж є використання інструментів візуалізації:

- побудова графіків втрат та інших метрик продуктивності,
- візуалізація обчислювальних графіків,
- показ додаткових даних під час навчання мережі.

Пакетний градієнтний спуск

GD

Пакетний градієнтний спуск

Щоб мінімізувати $\mathcal{J}(W)$, **пакетний градієнтний спуск** (GD) використовує наступне правило:

$$g_t = \frac{1}{n} \sum_{i=1}^n \nabla_W \mathcal{L} \left(y^{(i)}, f(\mathbf{x}^{(i)}, W) \right) = \nabla_W \mathcal{J}(W)$$

$$W_{t+1} = W_t - \alpha g_t,$$

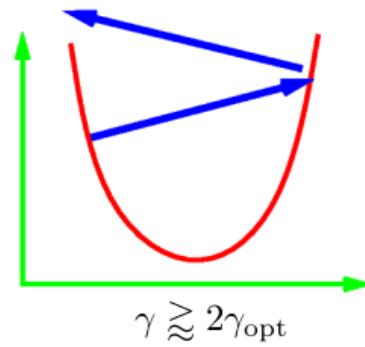
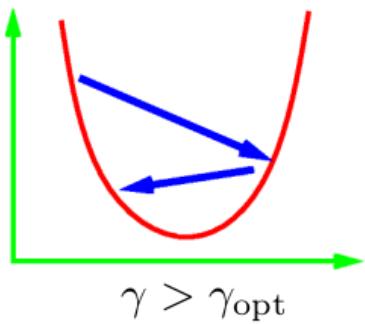
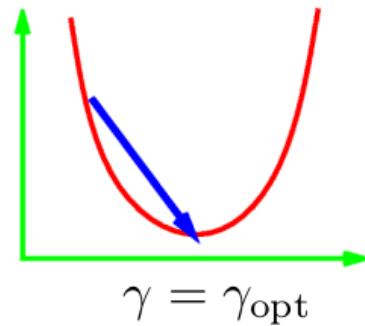
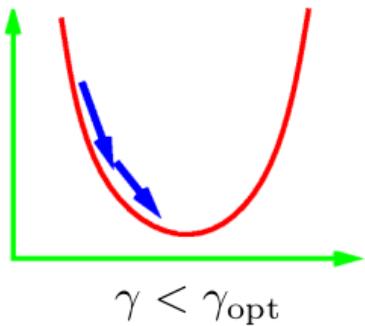
дек α – крок навчання.



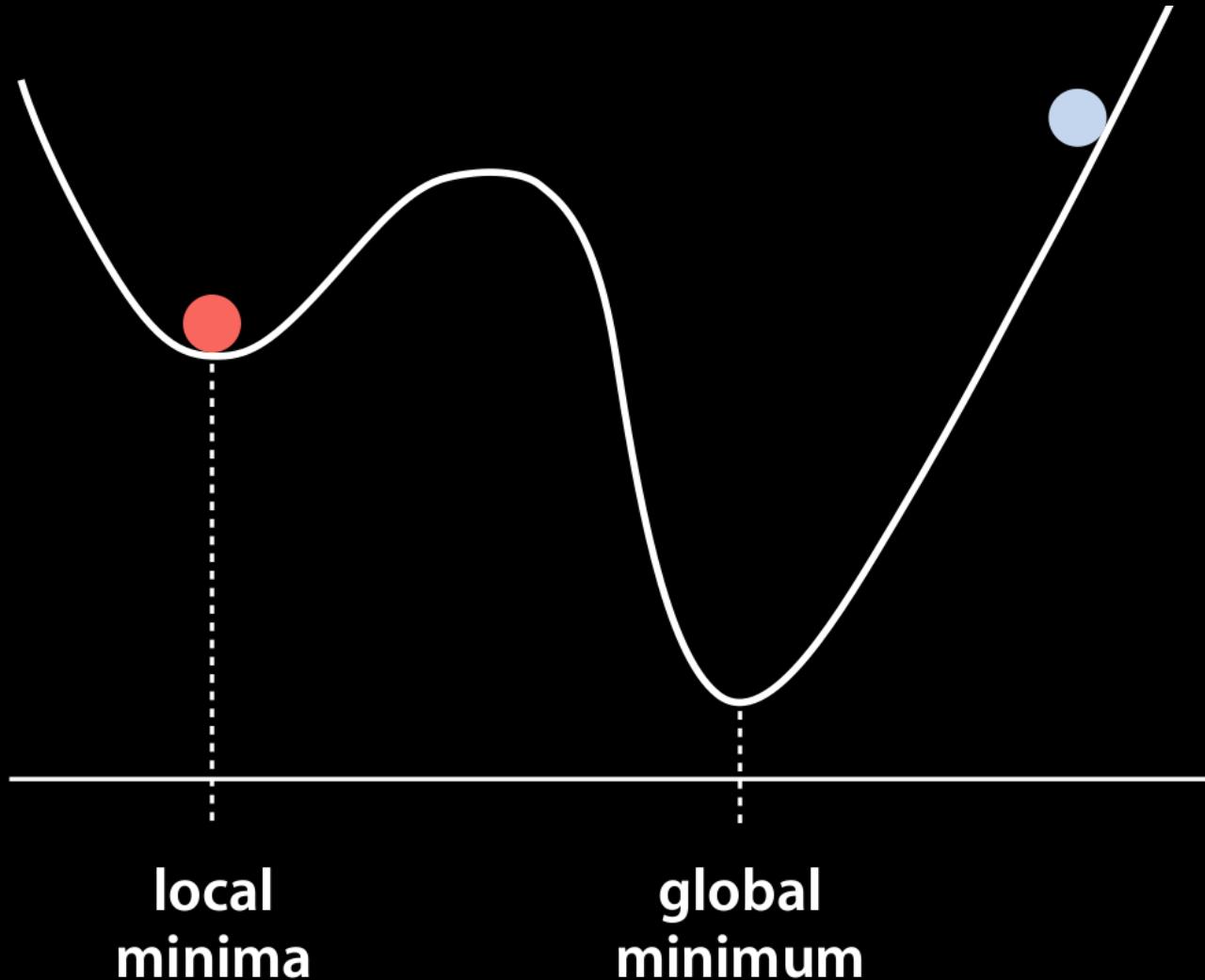
Найгірший метод оптимізації в світі

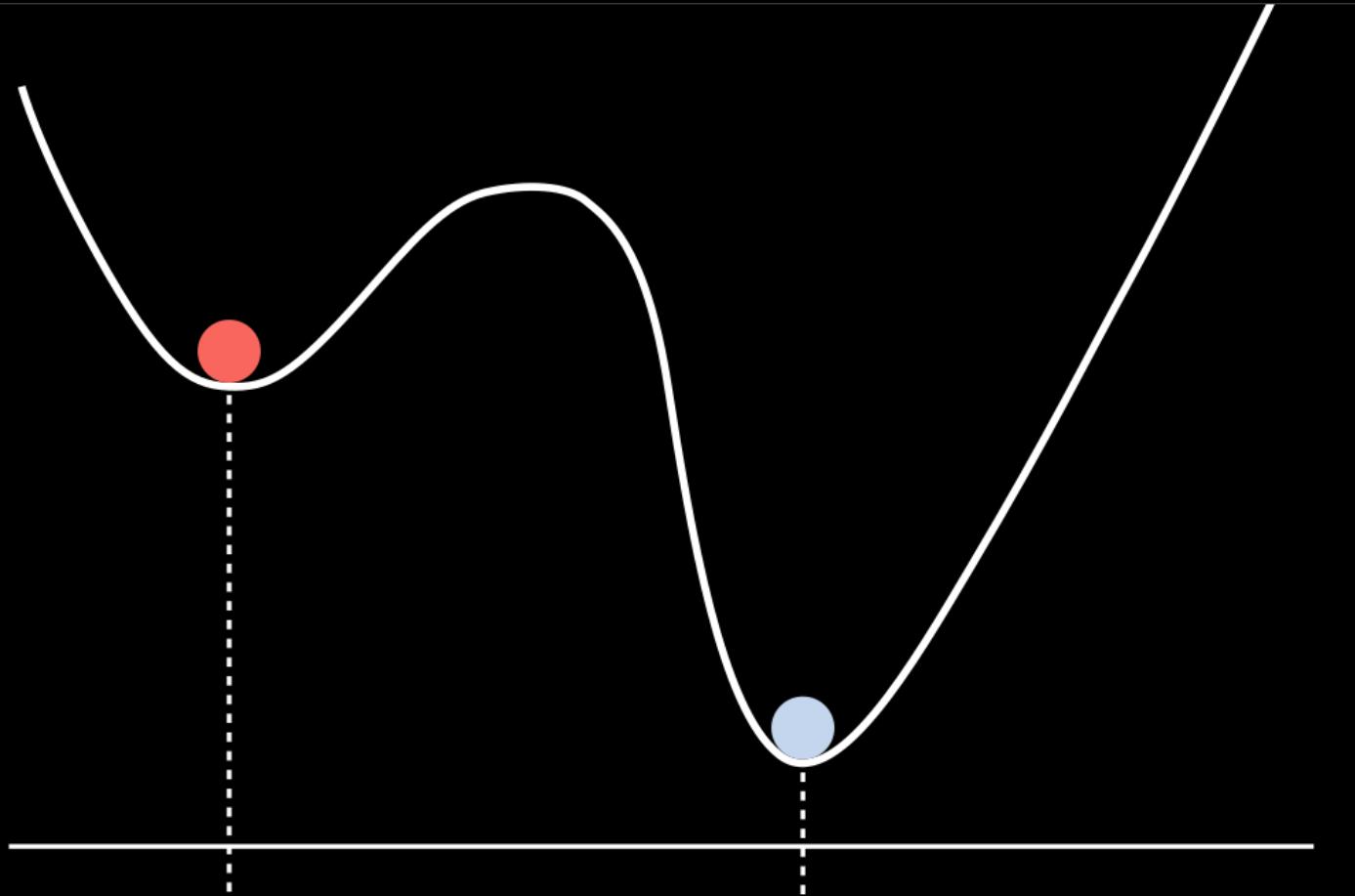
Примітка: ймовірно, Вам ніколи не слід використовувати градієнтний спуск безпосередньо, вважайте це будівельним блоком для інших методів.

Крок навчання



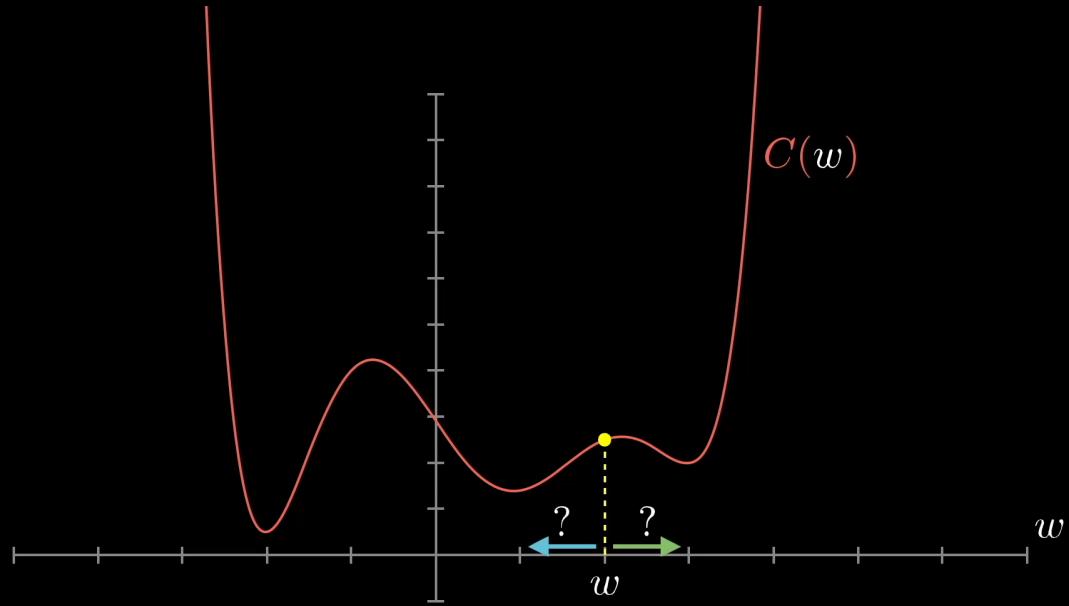
Тут $\gamma = \alpha$ – крок навчання.





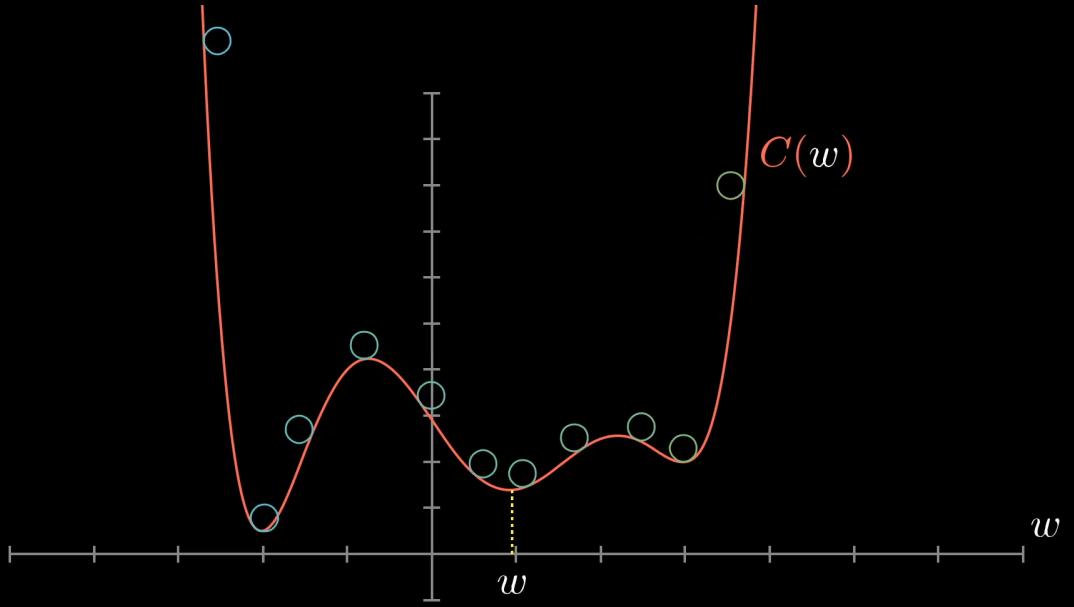
local
minima

global
minimum



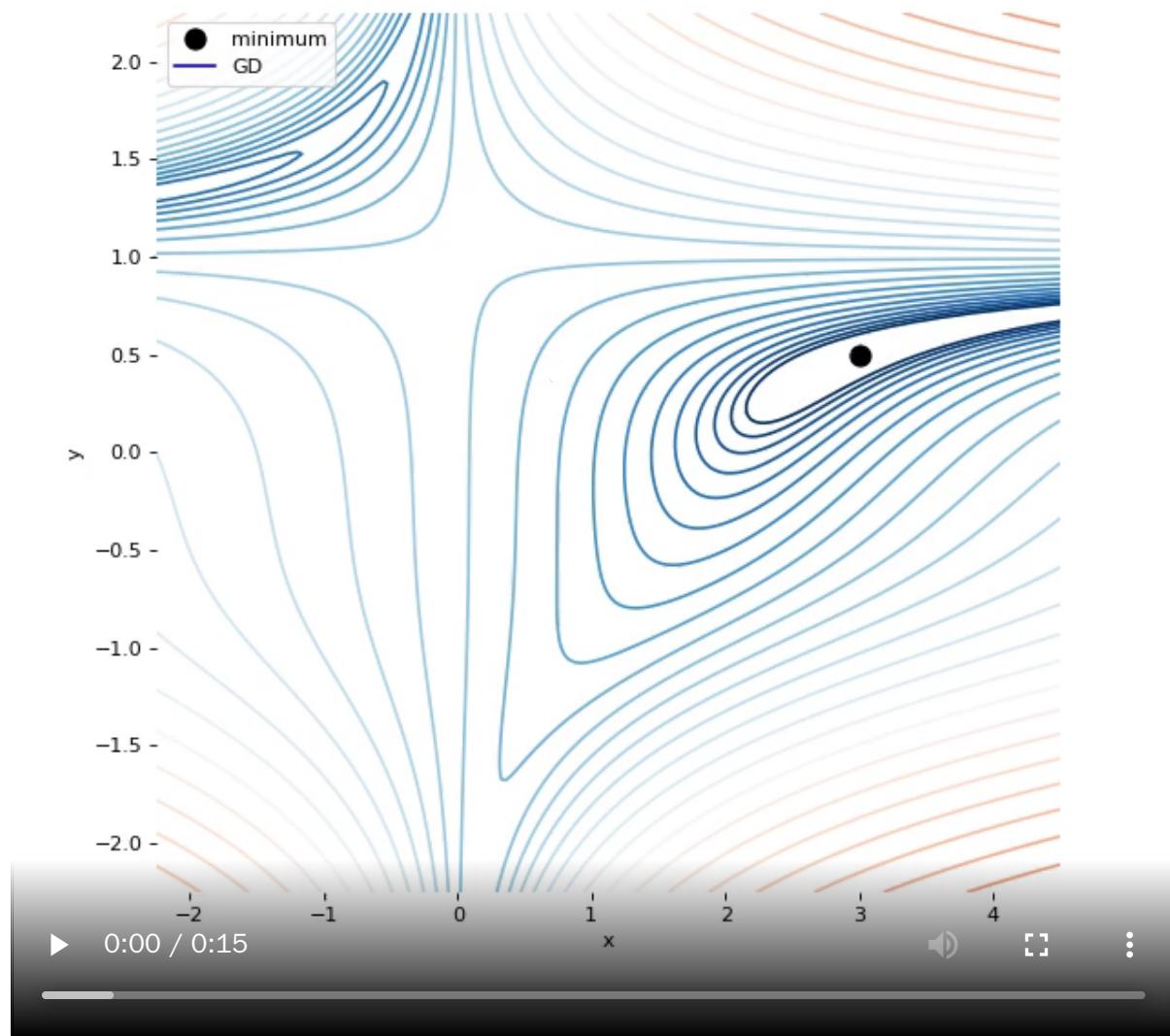
▶ 0:00 / 0:08

🔊 [] ⋮



▶ 0:00 / 0:04

🔊 [] ⋮



Стохастичний градієнтний спуск

SGD

SGD

Щоб зменшити обчислювальну складність, **стохастичний градієнтний спуск** (SGD) полягає в оновленні параметрів після кожного прикладу

$$\ell^{(i)} = \mathcal{L} \left(y^{(i)}, f(\mathbf{x}^{(i)}, W) \right)$$

$$g_t^{(i)} = \nabla_W \ell^{(i)}$$

$$W_{t+1} = W_t - \alpha g_t^{(i)}$$

Переваги SGD

- Стохастичні градієнти обчислювати значно легше (пропорційно розміру набору даних), тому Ви часто можете зробити тисячі кроків SGD за вартість одного кроку GD.
- У середньому стохастичний градієнт є хорошою оцінкою градієнта.
- Шум може перешкоджати оптимізаційному сходженню до поганих локальних мінімумів.

Міні-пакети

Міні-пакети

Обчислення втрат для міні-пакетів та оновлення параметрів

$$g_t^{(k)} = \frac{1}{B} \sum_{i=1}^B \nabla_W \mathcal{L} \left(y_k^{(i)}, f(\mathbf{x}_k^{(i)}, W) \right)$$
$$W_{t+1} = W_t - \alpha g_t^{(k)},$$

де k – індекс міні-пакета.

- Збільшення розміру пакету B зменшує дисперсію оцінок градієнта та забезпечує прискорення пакетної обробки.
- Взаємозв'язок між B і α все ще незрозумілий.

Импульс

Імпульс

SGD + Імпульс = Стохастичний метод важкої кулі

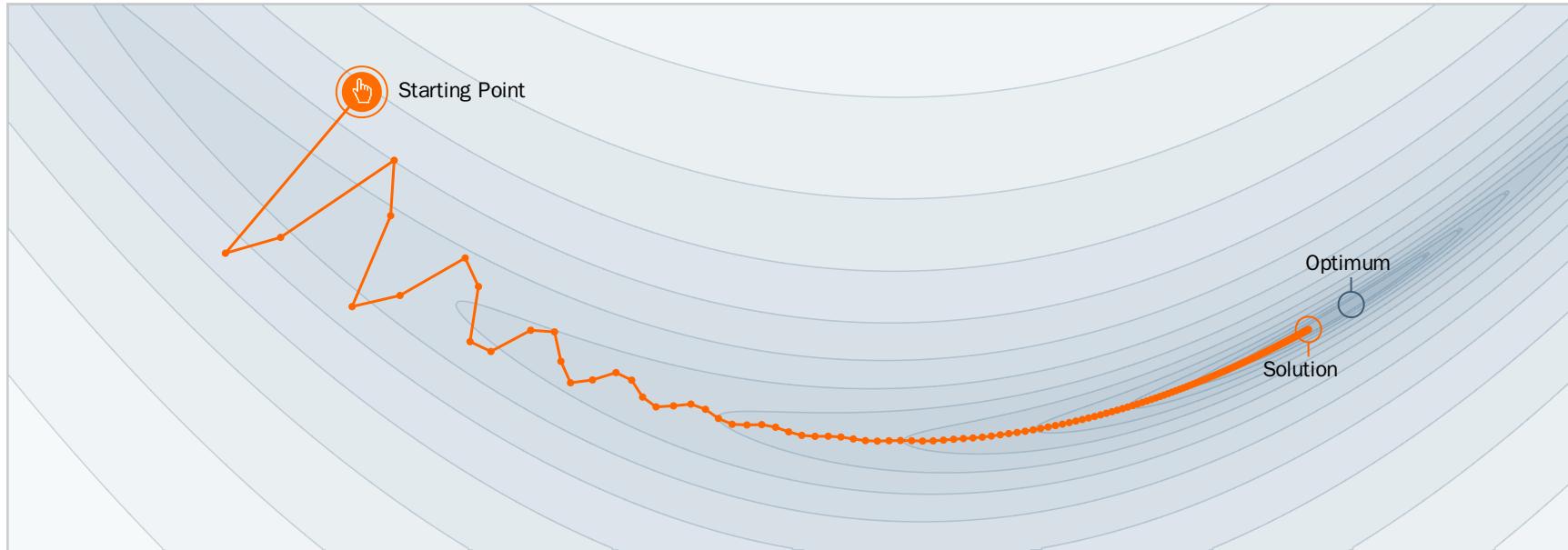
$$p_{t+1} = \beta_t p_t + \nabla \ell^{(i)}(W_t)$$

$$W_{t+1} = W_t - \alpha_t p_{t+1}$$

Правило оновлення:

$$W_{t+1} = W_t - \alpha_t \nabla \ell^{(i)}(W_t) + \beta_t (W_t - W_{t-1})$$

Ключова ідея: Наступний крок стає комбінацією напрямку попереднього кроку та нового негативного градієнта.



Step-size $\alpha = 0.02$



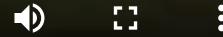
Momentum $\beta = 0.99$



We often think of Momentum as a means of dampening oscillations, speeding up the iterations, leading to faster convergence. But it creates its own oscillations. What is going on?

SGD with Momentum

▶ 0:00 / 0:46



Переваги

SGD з імпульсом має **три** хороші властивості:

- він може пройти через локальні бар'єри
- прискорюється, якщо градієнт не сильно змінюється
- гасить коливання у вузьких долинах

Практичні аспекти імпульсу

По суті, це «**безкоштовний обід**», майже в усіх ситуаціях **SGD + імпульс** краще, ніж SGD, і дуже рідко гірше!

Рекомендовані параметри:

$\beta = 0.9$ or 0.99 майже завжди працють добре. Іноді можна отримати невеликі переваги, налаштувавши його.

Параметр розміру кроку (α) зазвичай потрібно зменшувати, коли параметр імпульсу збільшується, щоб підтримувати збіжність.

